中置運算式轉換為前置表示法

資訊四丙

D0683497

柯利韋

目錄

[一、 資料結構 2](#_Toc56780907)

[1. 數值或是符號 2](#_Toc56780908)

[2. 符號優先權 2](#_Toc56780909)

[3. 字元資料 2](#_Toc56780910)

[4. 堆疊 3](#_Toc56780911)

[5. 運算式 5](#_Toc56780912)

[二、 演算法說明 6](#_Toc56780913)

[三、 程式原始碼 7](#_Toc56780914)

[1. lib.h 7](#_Toc56780915)

[2. D0683497.cpp 10](#_Toc56780916)

[四、 執行結果 15](#_Toc56780917)

[五、 心得 15](#_Toc56780918)

1. 資料結構
   1. 數值或是符號

enum Type

{

val,

oper

};

* 1. 符號優先權

enum Precedence

{

add = 1,

sub = 1,

mult = 2,

division = 2,

mod = 2,

left = 0,

right = 0

};

* 1. 字元資料

typedef struct Data

{

Type type;

int value;

char oper;

Precedence pre;

} Data;

* 1. 堆疊

typedef struct Stack

{

int top;

Data data[MAX\_SIZE];

bool IsEmpty();

bool IsFull();

void Push(Data d);

Data Pop();

Stack()

{

top=-1;

}

} Stack;

bool Stack::IsEmpty()

{

if(top == -1)

return true;

else

return false;

}

bool Stack::IsFull()

{

if(top == MAX\_SIZE-1)

return true;

else

return false;

}

void Stack::Push(Data d)

{

if(IsFull() == true)

printf("Stack is full. Cannot Push any element.\n");

else

{

top++;

data[top]= d;

}

}

Data Stack::Pop()

{

if(IsEmpty() == true)

{

printf("Stack is Empty. Cannot pop any element.\n");

}

else

{

return data[top--];

}

}

* 1. 運算式

typedef struct Order

{

int inCount;

Data in[MAX\_SIZE]; // 原資料

int preCount;

Data pre[MAX\_SIZE]; // 前序

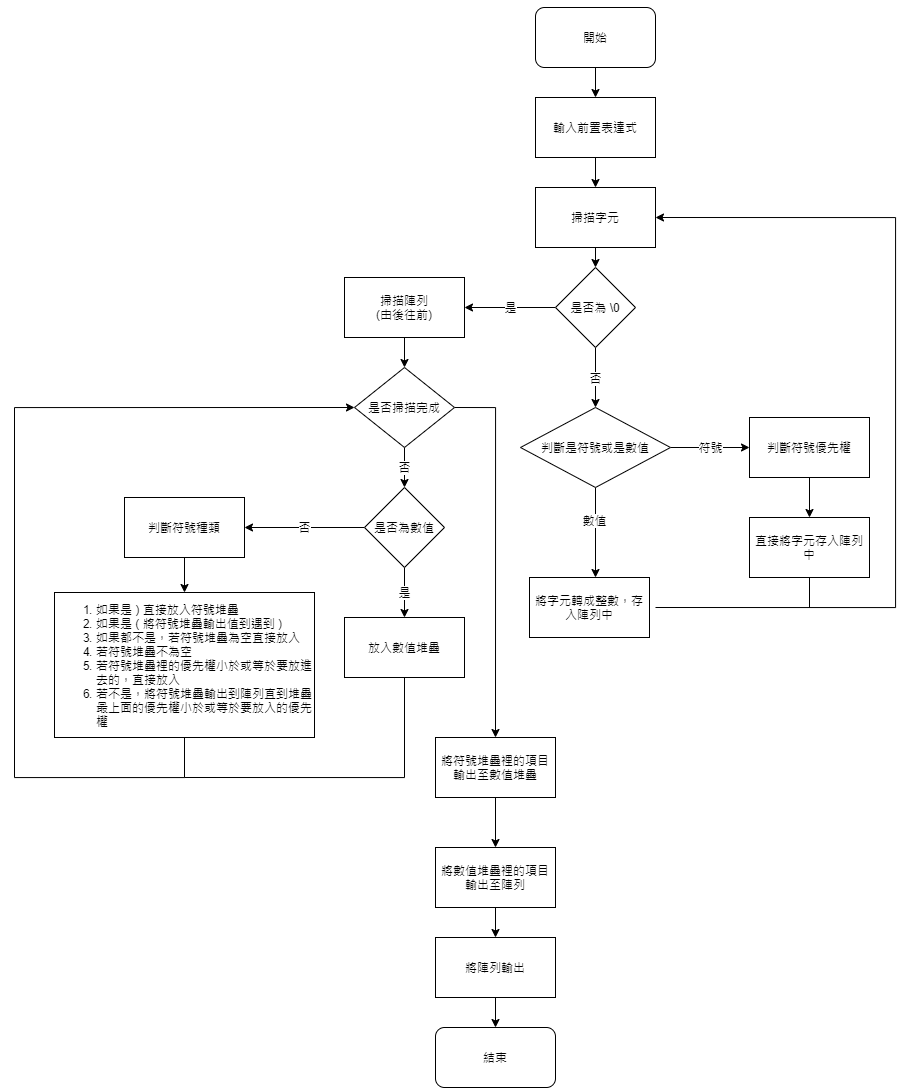
void Set(char\* string);

void InToPre();

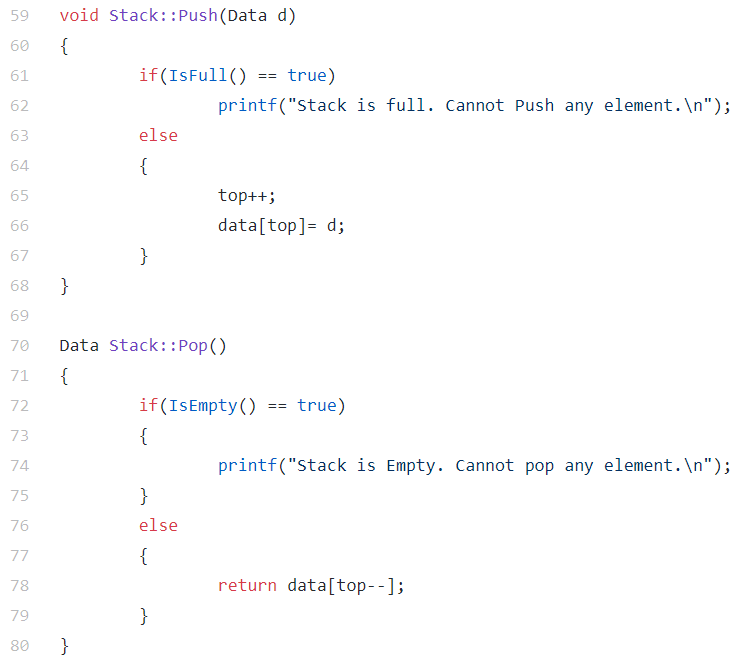
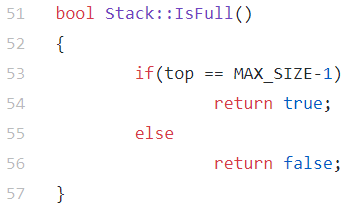
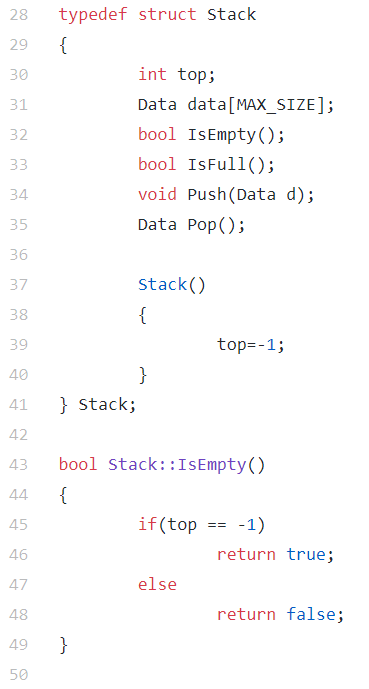
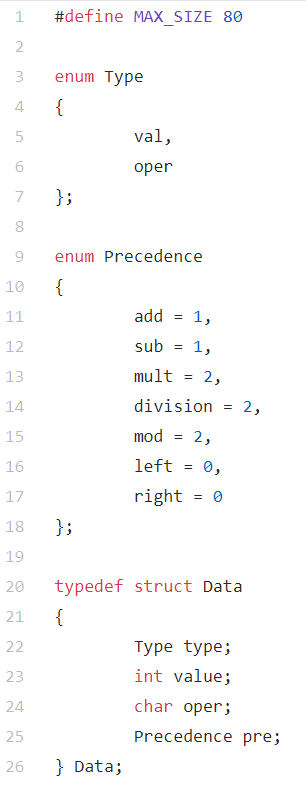
void ShowPre();

} Order;

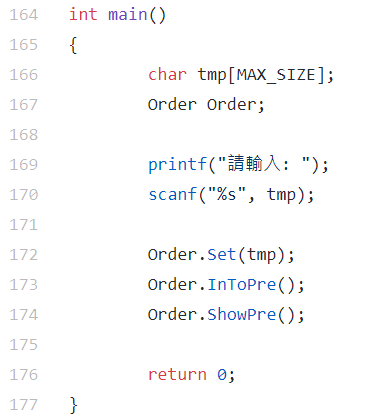
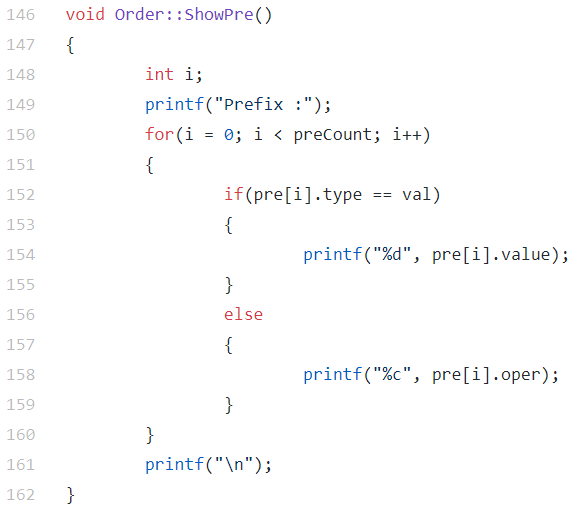
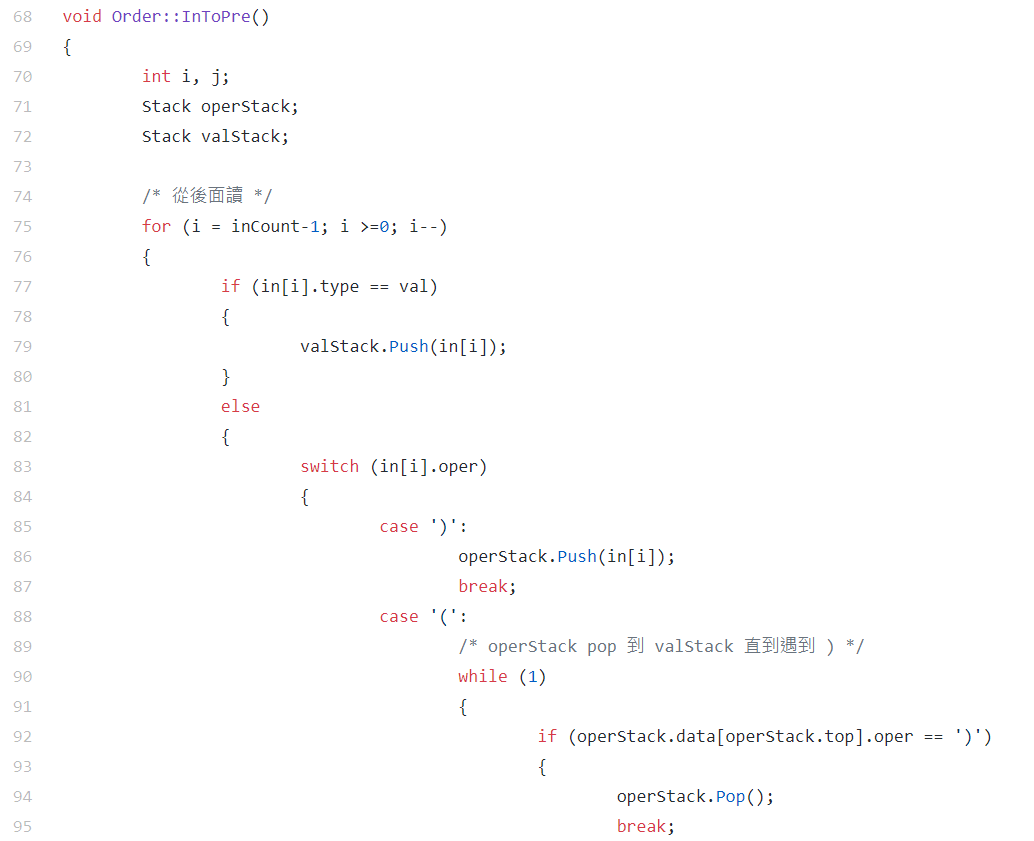
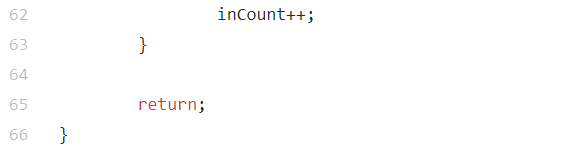
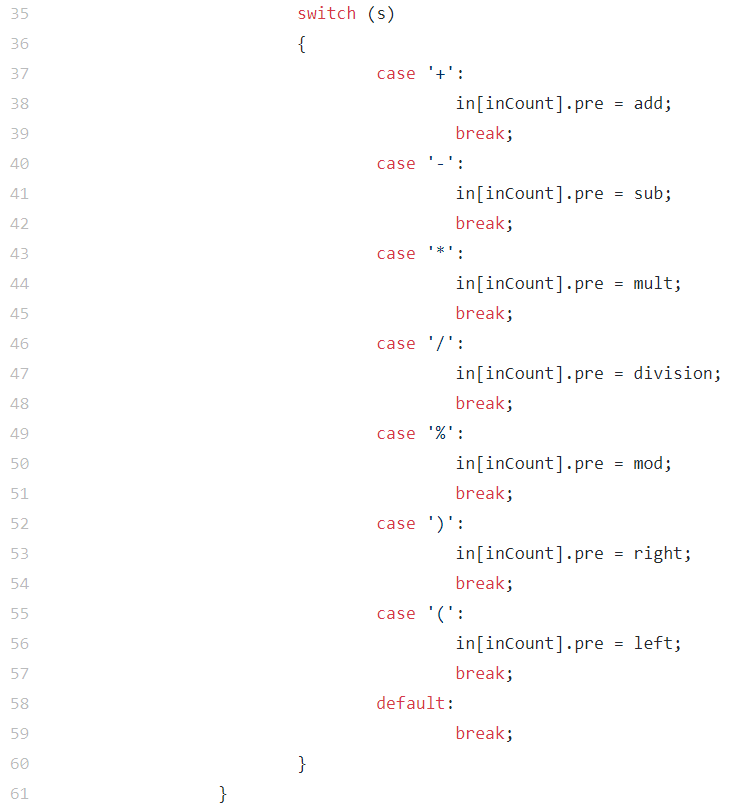
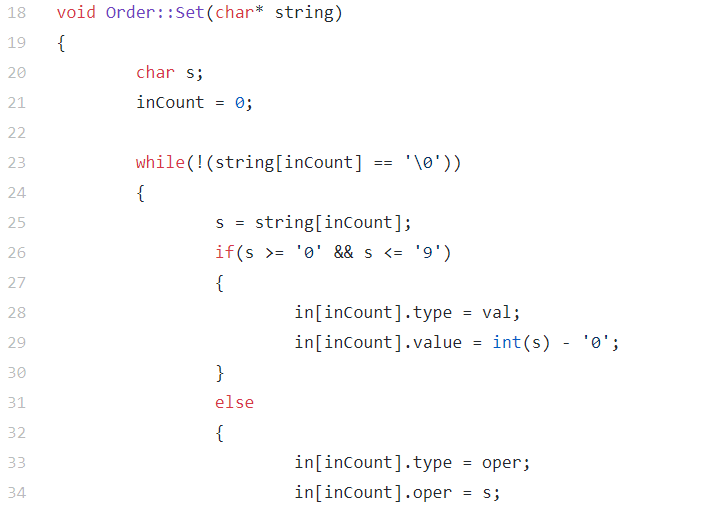
1. 演算法說明



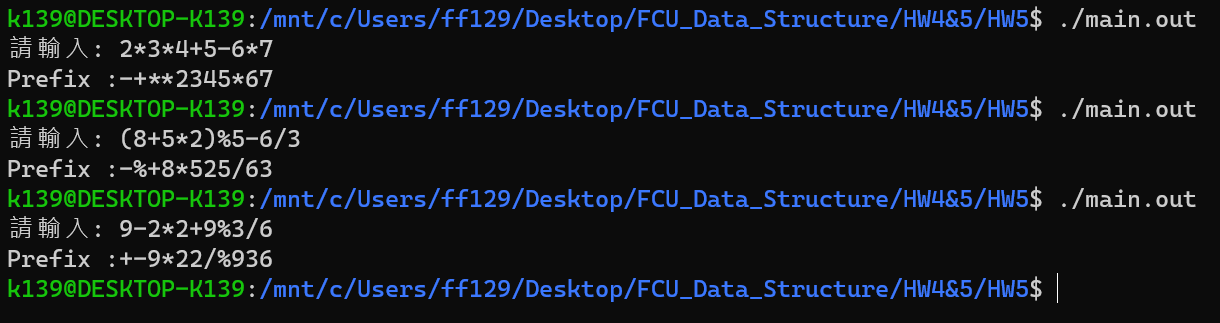
1. 程式原始碼
   1. lib.h



* 1. D0683497.cpp



1. 執行結果



1. 心得

基本上實習課已經練習過蠻多次的了，所以大致上沒有什麼問題。